UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/042,581	01/09/2002	Thanh V. Lam	POU920010124US1	6963

23334 7590 04/29/2005

FLEIT, KAIN, GIBBONS, GUTMAN, BONGINI
& BIANCO P.L.
ONE BOCA COMMERCE CENTER
551 NORTHWEST 77TH STREET, SUITE 111
BOCA RATON, FL 33487

EXAMINER	
TAN, ALVIN H	
ART UNIT	PAPER NUMBER
2173	

DATE MAILED: 04/29/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	**Application No.** 10/042,581	**Applicant(s)** LAM ET AL.	
	Examiner Alvin H. Tan	**Art Unit** 2173	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *1/9/02*.

2a)□ This action is **FINAL**. 2b)☒ This action is non-final.

3)□ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-24* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)□ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-24* is/are rejected.

7)□ Claim(s) _____ is/are objected to.

8)□ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *09 January 2002* is/are: a)□ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)□ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)□ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)□ All b)□ Some * c)□ None of:

1.□ Certified copies of the priority documents have been received.

2.□ Certified copies of the priority documents have been received in Application No. _____.

3.□ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) □ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date *1/9/02*.

4) □ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ .

5) □ Notice of Informal Patent Application (PTO-152)

6) □ Other: _____.

DETAILED ACTION

Remarks

1. Claims 1-24 have been examined and rejected. This document is the first Office action on the merits.

Drawings

2. The drawings are objected to as failing to comply with 37 CFR 1.84(p)(4) because reference character "232" has been used to designate both the "Do All" and "Skip" function buttons. Corrected drawing sheets in compliance with 37 CFR 1.121(d) are required in reply to the Office action to avoid abandonment of the application. Any amended replacement drawing sheet should include all of the figures appearing on the immediate prior version of the sheet, even if only one figure is being amended. Each drawing sheet submitted after the filing date of an application must be labeled in the top margin as either "Replacement Sheet" or "New Sheet" pursuant to 37 CFR 1.121(d). If the changes are not accepted by the examiner, the applicant will be notified and informed of any required corrective action in the next Office action. The objection to the drawings will not be held in abeyance.

Specification

3. The disclosure is objected to because of the following informalities:

a. *[On page 4, line 18]* examiner suggests rewording "... allows the operator to accept accepting a selection of at least one of the commands" for clarity.

b. *[On page 4, line 22]* examiner suggests changing "an system" to --a system--.

c. *[On page 6, line 4]* examiner suggests changing "will result execution" to --will result in execution--.

d. *[On page 6, line 21]*, applicant incorrectly refers to the local computer with reference character "106". Examiner suggests changing reference character "106" to reference character "108".

e. *[On page 10, lines 3,6]* applicant incorrectly refers to the command window with reference character "102". Examiner suggest changing reference character "102" to reference character "204".

f. *[On page 11, line 23]* examiner suggests changing "to causes" to --to cause--.

g. *[On page 12, lines 18-19]* examiner suggests changing "group of command" to --group of commands--.

h. Reference character "232" has been used to reference two different parts in the specification. It has been used in reference to the "Do All" function button *[page 11, lines 22,30; page 12, line 16; page 13, lines 20,25; page 14, lines 21,27]* and the "Skip" function button *[page 15, lines 22-23, 26; page 16, line 1; page 17, line 6]*.

Appropriate correction is required.

Claim Objections

4. Claims 9-16, with respect to claim 9, and claim 13 is objected to because of the following informalities:

a. *[On line 10]* of claim 9, examiner suggests changing "at lest one" to --at least one--.

b. *[On line 4]* of claim 13, examiner suggests changing "in association the" to --in association with the--.

Appropriate correction is required.

Claim Rejections - 35 USC § 102

5. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

6. Claims 1-4, 9-12, 17-20 are rejected under 35 U.S.C. 102(e) as being anticipated by Maddocks et al (Pub No. US 2004/0201627 A1).

Claims 1-4

6-1. Regarding claim 1, Maddocks anticipates the claimed invention by disclosing a "tree-based GUI that enables test programs for testing various devices" *[paragraph 15, lines 1-3]*. Maddocks anticipates the claim for a method comprising accepting an operator defined plurality of computer operating system commands and displaying the plurality of computer operating system commands, by teaching, "All of the information needed to enable tests to be set up and executed may be stored in a memory device comprised by a computer" *[paragraph 17, lines 1-3]*. This would "enable multiple users to create, edit, and/or execute machine control sequences at their workstations by accessing previously created machine control sequences" *[paragraph 17, lines 11-14]*. Since all of the information needed to enable tests is stored, users are able to retrieve any part of the information needed to create their own tests. Hence, the commands associated with the devices when the device node is selected *[paragraph 22, lines 1-3]* are defined by the user. In this way, the GUI accepts an operator defined plurality of computer operating system commands. The list of commands is displayed as shown *[figure 3, reference character "35"]* when the user selects a device node by clicking the left mouse button once *[paragraph 22, lines 1-3]*.

Maddocks anticipates the method comprising accepting a selection of at least one computer operating system command from within the plurality of computer operating system commands, by teaching that the user can select commands and add them to the tree by highlighting the selected command from the Jlist "35" *[figure 3]* and then using the "ADD" button *[paragraph 22, lines 4-5]*.

Maddocks anticipates the method comprising executing the at least one computer operating system command contained within the selection, by teaching that the selected commands are executed when the user selects the "Run Sequence" tab and then selects the machine control sequence to be executed *[paragraph 28, lines 1-4]*.

6-2. Regarding claim 2, Maddocks anticipates the method wherein the operator defined plurality of computer operating system commands is contained within a data file, by teaching that "all of the information needed to enable tests to be set up and executed may be stored in a memory device comprised by a computer. Alternatively, some or all of the needed information could be stored in a memory device that functions as a library or repository for command sequences" *[paragraph 17, lines 1-6]*. This would "enable multiple users to create, edit, and/or execute machine control sequences at their workstations by accessing previously created machine control sequences and/or the corresponding results over the network" *[paragraph 17, lines 12-15]*. Thus, the operator defined plurality of commands must be stored in some sort of data file in order for other users to access it.

6-3. Regarding claim 3, Maddocks anticipates the method wherein executing comprises single stepping through each computer operating system command within the selection, by teaching, "The Jtree is a hierarchical tree structure of sequences, steps, devices, and commands" *[paragraph 19, lines 9-10]*. Maddocks also teaches "a particular machine control sequence is executed when the user selects the "Run Sequence" tab shown *[figure 3]* and then selects the machine control sequence to be executed. Thus, a user can single step through each command within the selection of commands if the user were to create multiple sequences, each with a single step, device, and command selected from the plurality of commands in Jlist "35" *[figure 3]*, and then execute each sequence.

6-4. Regarding claim 4, Maddocks anticipates the method of displaying a program output and an error output within separate displays, wherein the separate displays are at least one of separate GUI windows and separate display screens, by teaching, "Once the "Sequence Results" window "72" has been opened, an item "76" and value "77" *[figure 6]* pair are displayed. Underneath this pair, a number of statistics are displayed that provide the user with a high-level summary of the execution results of the sequence for the entire sequence run" *[paragraph 30, lines 1-5]*. This summary of execution results represents the program output. "Whether the command succeeded or failed is indicated under labels "92" and "93" *[figure 6]* in the line in which the corresponding command is shown" *[paragraph 32, lines 4-6]*. This is where the error output is displayed.

Maddocks teaches "The "Sequence Results Viewer" portion of the GUI is a multiple document interface (MDI), which means that it allows multiple set of results, i.e., multiple files, to be simultaneously displayed in different "Sequence Results" windows *[paragraph 29, lines 1-5]*. The sequence results windows "72" and "75" *[figure*

6] may either be displayed as separate windows or as active portions of a different window so that they are simultaneously and fully viewable by the user *[paragraph 29, lines 22-25]*. Thus, the user may be able to open two separate GUI windows with the same sequence results and view the program output in one of them and the error output in the other.

Claims 9-12

6-5. Regarding claim 9, Maddocks anticipates a system comprising a memory for storing a computer operating system command list, by teaching, "All of the information needed to enable tests to be set up and executed may be stored in a memory device comprised by a computer. Alternatively, some or all of the needed information could be stored in a memory device that functions as a library or repository for command sequences" *[paragraph 17, lines 1-6]*. Thus, a command list "35" *[figure 3]* can be stored in memory.

Maddocks anticipates the system comprising a command display, communicatively coupled to the memory, for displaying at least one computer operating system command of the computer operating system command list. *[Figure 1]* shows a computer connected to the library by a network. The list of commands is displayed as shown *[figure 3, reference character "35"]* when the user selects a device node by clicking the left mouse button once *[paragraph 22, lines 1-3]*.

Maddocks anticipates the system comprising a command selector, communicatively coupled to the memory and the command display, for selecting at least one computer operating system command contained within the computer operating system command list, by teaching that the user can select commands and add them to the tree by highlighting the selected command from the Jlist "35" *[figure 3]* and then using the "ADD" button *[paragraph 22, lines 4-5]*.

Maddocks anticipates the system comprising a command dispatcher, communicatively coupled to the command selector, for causing execution of the selected at least one computer operating system command contained within the computer operating system command list, by teaching that the selected commands are executed when the user selects the "Run Sequence" tab and then selects the machine control sequence to be executed *[paragraph 28, lines 1-4]*.

6-6. Regarding claim 10, Maddocks anticipates the system wherein the computer operating system command list is stored within a data file in the memory, by teaching that "all of the information needed to enable tests to be set up and executed may be stored in a memory device comprised by a computer. Alternatively, some or all of the needed information could be stored in a memory device that functions as a library or repository for command sequences" *[paragraph 17, lines 1-6]*. This would "enable multiple users to create, edit, and/or execute machine control sequences at their workstations by accessing previously created machine control sequences and/or the corresponding results over the network" *[paragraph 17, lines 12-15]*. Thus, the operator defined plurality of commands must be stored in some sort of data file in order for other users to access it.

6-7. Regarding claim 11, Maddocks anticipates the system wherein the command dispatcher further performs single stepping through each command within the selected computer operating system command list, by teaching, "The Jtree is a hierarchical tree structure of sequences, steps, devices, and commands" *[paragraph 19, lines 9-10]*. Maddocks also teaches "a particular machine control sequence is executed when the user selects the "Run Sequence" tab shown *[figure 3]* and then selects the machine control sequence to be executed. Thus, the user can single step through each command within the selection of commands if the user were to create multiple sequences, each with a single step, device, and command selected from the plurality of commands in Jlist "35" *[figure 3]*, and then execute each sequence.

6-8. Regarding claim 12, Maddocks anticipates the system further comprising a program standard output display and a standard error output display that are each separate displays, wherein the separate displays are at least one of separate GUI windows and separate display screens, by teaching, "Once the "Sequence Results" window "72" has been opened, an item "76" and value "77" *[figure 6]* pair are displayed. Underneath this pair, a number of statistics are displayed that provide the user with a high-level summary of the execution results of the sequence for the entire sequence run" *[paragraph 30, lines 1-5]*. This summary of execution results represents the program output. "Whether the command succeeded or failed is indicated under labels

"92" and "93" *[figure 6]* in the line in which the corresponding command is shown" *[paragraph 32, lines 4-6]*. This is where the error output is displayed.

Maddocks teaches "The "Sequence Results Viewer" portion of the GUI is a multiple document interface (MDI), which means that it allows multiple set of results, i.e., multiple files, to be simultaneously displayed in different "Sequence Results" windows *[paragraph 29, lines 1-5]*. The sequence results windows "72" and "75" *[figure 6]* may either be displayed as separate windows or as active portions of a different window so that they are simultaneously and fully viewable by the user *[paragraph 29, lines 22-25]*. Thus, the user may be able to open two separate GUI windows with the same sequence results and view the program output in one of them and the error output in the other.

Claims 17-20

6-9. Regarding claim 17, Maddocks anticipates the claimed invention by disclosing a "tree-based GUI that enables test programs for testing various devices" *[paragraph 15, lines 1-3]*. Maddocks anticipates the claim for a computer readable medium including computer instructions for controlling and monitoring computer command execution, the computer instructions comprising instructions for accepting an operator defined plurality of computer operating system commands and displaying the plurality of computer operating system commands, by teaching, "All of the information needed to enable tests to be set up and executed may be stored in a memory device comprised by a computer" *[paragraph 17, lines 1-3]*. This would "enable multiple users to create, edit, and/or execute machine control sequences at their workstations by accessing previously created machine control sequences" *[paragraph 17, lines 11-14]*. Since all of the information needed to enable tests is stored, users are able to retrieve any part of the information needed to create their own tests. Hence, the commands associated with the devices when the device node is selected *[paragraph 22, lines 1-3]* are defined by the user. In this way, the GUI accepts an operator defined plurality of computer operating system commands. The list of commands is displayed as shown *[figure 3, reference character "35"]* when the user selects a device node by clicking the left mouse button once *[paragraph 22, lines 1-3]*.

Maddocks anticipates computer instructions for accepting a selection of at least one computer operating system command from within the plurality of computer operating system commands, by teaching that the user can select commands and add them to the tree by highlighting the selected command from the Jlist "35" *[figure 3]* and then using the "ADD" button *[paragraph 22; lines 4-5]*.

Maddocks anticipates computer instructions for executing the at least one computer operating system command contained within the selection, by teaching that the selected commands are executed when the user selects the "Run Sequence" tab and then selects the machine control sequence to be executed *[paragraph 28, lines 1-4]*.

6-10. Regarding claim 18, Maddocks anticipates the computer readable medium wherein the operator defined plurality of computer operating system commands is contained within a data file, by teaching that “all of the information needed to enable tests to be set up and executed may be stored in a memory device comprised by a computer. Alternatively, some or all of the needed information could be stored in a memory device that functions as a library or repository for command sequences” *[paragraph 17, lines 1-6]*. This would “enable multiple users to create, edit, and/or execute machine control sequences at their workstations by accessing previously created machine control sequences and/or the corresponding results over the network” *[paragraph 17, lines 12-15]*. Thus, the operator defined plurality of commands must be stored in some sort of data file in order for other users to access it.

6-11. Regarding claim 19, Maddocks anticipates the computer readable medium wherein the instructions for executing comprises single stepping through each computer operating system command within the selection, by teaching, “The Jtree is a hierarchical tree structure of sequences, steps, devices, and commands” *[paragraph 19, lines 9-10]*. Maddocks also teaches “a particular machine control sequence is executed when the user selects the “Run Sequence” tab shown *[figure 3]* and then selects the machine control sequence to be executed. Thus, a user can single step through each command within the selection of commands if the user were to create multiple sequences, each with a single step, device, and command selected from the plurality of commands in Jlist “35” *[figure 3]*, and then execute each sequence.

6-12. Regarding claim 20, Maddocks anticipates the computer readable medium further including instructions for displaying a program output and an error output within separate displays, wherein the separate displays are at least one of separate GUI windows and separate display screens, by teaching, "Once the "Sequence Results" window "72" has been opened, an item "76" and value "77" *[figure 6]* pair are displayed. Underneath this pair, a number of statistics are displayed that provide the user with a high-level summary of the execution results of the sequence for the entire sequence run" *[paragraph 30, lines 1-5]*. This summary of execution results represents the program output. "Whether the command succeeded or failed is indicated under labels "92" and "93" *[figure 6]* in the line in which the corresponding command is shown" *[paragraph 32, lines 4-6]*. This is where the error output is displayed.

Maddocks teaches "The "Sequence Results Viewer" portion of the GUI is a multiple document interface (MDI), which means that it allows multiple set of results, i.e., multiple files, to be simultaneously displayed in different "Sequence Results" windows *[paragraph 29, lines 1-5]*. The sequence results windows "72" and "75" *[figure 6]* may either be displayed as separate windows or as active portions of a different window so that they are simultaneously and fully viewable by the user *[paragraph 29, lines 22-25]*. Thus, the user may be able to open two separate GUI windows with the same sequence results and view the program output in one of them and the error output in the other.

7. Claims 8 and 24 are rejected under 35 U.S.C. 102(e) as being anticipated by Foster et al (US Patent No 6,684,260).

Claim 8

7-1. Regarding claim 8, Foster anticipates the claimed invention by disclosing a method for maintaining consistency of device driver settings. Foster anticipates the method of defining a plurality of computer operating system commands and displaying them, by teaching a computer which in operation executes the instructions of an operating system, one or more application programs, and a device driver for the purpose of coordinating use of the peripheral device in accomplishing the purpose of one or more application programs *[column 5, lines 28-35]*. "Operating systems interface with peripheral devices indirectly through device drivers. The operating system developer defines a device driver interface between the operating system and the device driver" *[column 1, lines 29-33]*. Foster teaches that when an application program requests access to device settings, the device driver activates a client user interface and a user interface process. User interface process may receive default device settings from several sources. Regardless of the method by which the user interface process obtains current and/or default device settings, user interface process prepares a dialog box with appropriate controls and appropriate initial values of attributes described by the dialog box and then presents the dialog box via the GUI to the user *[column 9, lines 13-29]*. Thus, because the operating system interfaces with peripheral devices through device drivers, operating system commands are defined by the controls presented to the user in the interface.

Foster anticipates the method comprising associating each of at least one computer operating system command within the plurality of computer operating system commands with one of at least one undo command, by teaching, "If neither the "Cancel" *[figure 3, reference character 306]* nor the "OK" *[figure 3, reference character 310]* command button has been activated, the user input event is understood to include one or more modified or selected values for one or more attributes as indicated in the dialog box. At step "320" *[figure 3]* device driver "211" *[figure 1]* obtains from the message obtained at step "304" *[figure 3]* the identifier of the attribute(s) affected by the user input event. The current value(s) for each identified attribute is then obtained, and at step "322" *[figure 3]* is posted on undo list "226" *[figure 2]*. An identifier of the attribute to which each value is associated is also posted on undo list "226" *[figure 2]*' *[column 11, lines 5-14]*. Thus, the undo list has an undo command associated with the set of computer operating system commands represented by the modified or selected value.

Foster anticipates the method comprising displaying the at least one undo command along with an identification of an associated computer operating system command to which each of the at least one undo command is associated, by teaching a client user interface session including a unique instantiation of user interface process, validation process, undo list, review list, and client's settings *[column 7, lines 49-52]*. *[Figure 2]*, reference character "226" shows the undo list within the client user interface.

Foster anticipates the method comprising accepting and executing a selection of a selected undo command, by teaching in response to user input received at box "356" *[figure 5]*, one or more user-directed modifications will be reversed *[column 14, lines 27-28]*. "The contents of the undo list may include values for attributes which the user has modified, for example, as posted at step "322" *[figure 3]*; or, attributes and values as posted during operation of any selected rule, for example as posted at step "360" *[figure 5]*. At step "358" *[figure 5]* it is preferred to reinstate the attribute values that existed prior to receipt of the user event indicated at step "304" *[figure 3]*. Note that if the user input event at step "304" *[figure 3]* has been validated by a complete operation of step "340" *[figure 3]*, then the scope of the undo operation at step "358" *[figure 5]* corresponds to undoing one user input even cycle *[column 14, lines 32-42]*. Thus, the user can accept a selection of a selected undo command in response to step "356" *[figure 5]* whereby selecting cancel will execute the undo command.

Claim 24

7-2. Regarding claim 24, Foster anticipates the claimed invention by disclosing a method for maintaining consistency of device driver settings. Foster discloses his invention on a computer with a memory device, the memory device including indicia of instructions or statements which are executable or interpretable by one or more processors *[column 5, lines 10-16]*.

Foster anticipates the method of defining a plurality of computer operating system commands and displaying them, by teaching a computer which in operation executes the instructions of an operating system, one or more application programs, and a device driver for the purpose of coordinating use of the peripheral device in accomplishing the purpose of one or more application programs *[column 5, lines 28-35]*. "Operating systems interface with peripheral devices indirectly through device drivers. The operating system developer defines a device driver interface between the operating system and the device driver" *[column 1, lines 29-33]*. Foster teaches that when an application program requests access to device settings, the device driver activates a client user interface and a user interface process. User interface process may receive default device settings from several sources. Regardless of the method by which the user interface process obtains current and/or default device settings, user interface process prepares a dialog box with appropriate controls and appropriate initial values of attributes described by the dialog box and then presents the dialog box via the GUI to the user *[column 9, lines 13-29]*. Thus, because the operating system interfaces with peripheral devices through device drivers, operating system commands are defined by the controls presented to the user in the interface.

Foster anticipates the method comprising associating each of at least one computer operating system command within the plurality of computer operating system commands with one of at least one undo command, by teaching, "If neither the "Cancel" *[figure 3, reference character 306]* nor the "OK" *[figure 3, reference character 310]* command button has been activated, the user input event is understood to include one or more modified or selected values for one or more attributes as indicated in the dialog box. At step "320" *[figure 3]* device driver "211" *[figure 1]* obtains from the message obtained at step "304" *[figure 3]* the identifier of the attribute(s) affected by the user input event. The current value(s) for each identified attribute is then obtained, and at step "322" *[figure 3]* is posted on undo list "226" *[figure 2]*. An identifier of the attribute to which each value is associated is also posted on undo list "226" *[figure 2]*' *[column 11, lines 5-14]*. Thus, the undo list has an undo command associated with the set of computer operating system commands represented by the modified or selected value.

Foster anticipates the method comprising displaying the at least one undo command along with an identification of an associated computer operating system command to which each of the at least one undo command is associated, by teaching a client user interface session including a unique instantiation of user interface process, validation process, undo list, review list, and client's settings *[column 7, lines 49-52]*. *[Figure 2]*, reference character "226" shows the undo list within the client user interface.

Foster anticipates the method comprising accepting and executing a selection of a selected undo command, by teaching in response to user input received at box "356" *[figure 5]*, one or more user-directed modifications will be reversed *[column 14, lines 27-28]*. "The contents of the undo list may include values for attributes which the user has modified, for example, as posted at step "322" *[figure 3]*; or, attributes and values as posted during operation of any selected rule, for example as posted at step "360" *[figure 5]*. At step "358" *[figure 5]* it is preferred to reinstate the attribute values that existed prior to receipt of the user event indicated at step "304" *[figure 3]*. Note that if the user input event at step "304" *[figure 3]* has been validated by a complete operation of step "340" *[figure 3]*, then the scope of the undo operation at step "358" *[figure 5]* corresponds to undoing one user input even cycle *[column 14, lines 32-42]*. Thus, the user can accept a selection of a selected undo command in response to step "356" *[figure 5]* whereby selecting cancel will execute the undo command.

Claim Rejections - 35 USC § 103

8. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

9. Claims 5-7, 14-16, 21-23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Maddocks et al (Pub No US 2004/0201627 A1), supra, and Bonnell et al (US Patent No 5,655,081).

Claims 5-7

9-1. Regarding claim 5, Maddocks teaches the invention substantially as claimed. See section 6-1. Further, Maddocks teaches that the GUI generation code and the GUI user interaction handling code in his invention are written in the JAVA programming language to enable the present invention to be platform-independent and therefore portable over different computers, handheld devices and appliances, regardless of the operating system that is running on them *[paragraph 18, lines 1-6]*. Maddocks teaches, "All of the information needed to enable tests to be set up and executed may be stored in a memory device comprised by a computer" *[paragraph 17, lines 1-3]*. This would "enable multiple users to create, edit, and/or execute machine control sequences at their workstations by accessing previously created machine control sequences and/or the corresponding results over the network" *[paragraph 17, lines 11-15]*. Maddocks discloses that his invention is portable over many operating systems and that multiple users are able to access previously created tests.

While Maddocks teaches accessing machine control sequences over a network, Maddocks does not expressly teach a method wherein the executing is performed on a plurality of processors. Bonnell teaches, "Many business entities have one client/server network installed in each regional office, in which a high-capacity computer system operates as the "server" supporting many lower-capacity "client" desktop computers. The servers in such a business entity are also commonly connected to one another by a high-level network known as a wide area network. In this manner, users at any location within the business entity can theoretically access resources available in the company's network regardless of where the resource is located. The flexibility gained for users with this type of arrangement comes with a price, however. It is very difficult to manage such a diverse and widely-dispersed network for many reasons. Servers installed in the wide area network are frequently not all of the same variety. One regional office may be using an IBM machine with a UNIX operating system, while another regional office may be using a DEC machine with a VMS operating system" *[column 1, lines 27-44]*. Bonnell gives a solution to this problem by disclosing a method for managing a computer network *[figure 1]* where a network management computer system is coupled via a network to a server computer system and a plurality of other server computer systems. The hardware present in each of the computer systems need not be the same *[column 1, lines 54-61]*.

Since Maddocks' invention is portable over many operating systems and multiple users are able to access previously created tests, it would have been obvious to one of ordinary skill in the art at the time the invention was made to control the execution of commands, as taught by Maddocks, so that they would be executed on a plurality of processors as suggested by Bonnell, to allow for the execution of operating system commands to be performed on a plurality of processors. In this way, widely dispersed networks may be more easily managed by allowing the results of machine control sequence to be quickly and easily analyzed from a central console as suggested by Bonnell. Thus, businesses will get the benefit of allowing users at any location within the business entity to access resources available in the company's network regardless of where the resource is located while reducing the difficulty of having to manage a diverse network.

9-2. Regarding claim 6, Maddocks and Bonnell teach the invention substantially as claimed. See sections 6-1 and 9-1. Maddocks further teaches the method wherein executing comprises detecting a failure to execute the at least one computer operating system command, by teaching, "Whether the command succeeded or failed is indicated under labels "92" and "93" *[figure 6]* in the line in which the corresponding command is shown" *[paragraph 32, lines 4-6]*. Maddocks does not expressly teach the plurality of processors on which the detecting of a failure to execute a command is done on. Bonnell does teach the advantage of running a monitoring system on a plurality of computers as stated in section 9-1. Since Maddocks' invention is portable over many operating systems and multiple users are able to access previously created tests, it would have been obvious to one of ordinary skill in the art at the time the invention was made to allow for the detection of a failure to execute at least one computer operating system command, as taught by Maddocks, so that they would be detected on the plurality of processors as suggested by Bonnell, to allow for the detection of a failure to execute at least one operating system command to be performed on a plurality of processors. In this way, widely dispersed networks may be more easily managed by allowing the results of machine control sequence to be quickly and easily analyzed from a central console as suggested by Bonnell. Thus, businesses will get the benefit of allowing users at any location within the business entity to access resources available in the company's network regardless of where the resource is located while reducing the difficulty of having to manage a diverse network.

In addition, Maddocks does not expressly teach a method wherein the executing comprises determining at least one processor within the plurality of processors that is unavailable for executing the at least one computer operating system command. Bonnell teaches a flow diagram *[figure 9]* illustrating a procedure for monitoring resources according to the network management system *[column 8, lines 13-15]*. Reference character "144" *[figure 9]* shows that a manager software sends a signal to the agent to initiate a monitoring procedure. If the agent is unavailable, the procedure stops and the operating system commands will not execute *[reference character "152"]*. In this way, the manager software determines whether or not the agent is available for the execution of operating system commands. Bonnell also teaches that it is desirable in a large network to use numerous network management computer systems, each running its own manager software and to have agent processes in the network numbering in the thousands *[column 5, lines 37-41]*. Maddocks teaches that a need exists for a convenient and expedient way to view and analyze machine control sequences and the corresponding execution results *[paragraph 4, lines 1-3]*. By first sending a signal to the agent before execution begins, the system of Bonnell would be able to predetermine which agents are available for execution of commands and which are not. This would allow only the monitoring of agents that are available, which would result in a more organized and expedient method for analyzing machine control sequences over the network. It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Maddocks with the teachings of Bonnell so that the analysis of machine control sequences could be run on a plurality of processors wherein the system determines which processors are available before execution.

9-3. Regarding claim 7, Maddocks and Bonnell teach the invention substantially as claimed. See sections 6-1, 9-1, and 9-2. Bonnell teaches that a knowledge database of the manager software system *[figure 2, reference character "47"]* may contain setup commands *[figure 4, reference character "84"]*. "Setup commands are those that are to be executed whenever the manager software system establishes a connection with an agent software system" *[column 4, lines 21-23]*. Thus, when the manager software determines whether a processor is available or not, it is effectively requesting the execution of the commands. This claim shares the same motivation as was stated in section 9-2 above.

Claims 14-16

9-4. Regarding claim 14, Maddocks teaches the invention substantially as claimed (see section 6-5) and the claim is rejected for similar reasons as described in the rejection of claim 5 in section 9-1 above.

9-5. Regarding claim 15, Maddocks and Bonnell, teach the invention substantially as claimed (see sections 6-5 and 9-4) and the claim is rejected for similar reasons as described in the rejection of claim 6 in section 9-2 above.

9-6. Regarding claim 16, Maddocks and Bonnell, teach the invention substantially as claimed (see sections 6-5, 9-4, and 9-5) and the claim is rejected for similar reasons as described in the rejection of claim 7 in section 9-3 above.

Claims 21-23

9-7. Regarding claim 21, Maddocks teaches the invention substantially as claimed (see section 6-9) and the claim is rejected for similar reasons as described in the rejection of claim 5 in section 9-1 above.

9-8. Regarding claim 22, Maddocks and Bonnell, teaches the invention substantially as claimed (see sections 6-9 and 9-7) and the claim is rejected for similar reasons as described in the rejection of claim 6 in section 9-2 above.

9-9. Regarding claim 23, Maddocks and Bonnell, teaches the invention substantially as claimed (see sections 6-9, 9-7, and 9-8) and the claim is rejected for similar reasons as described in the rejection of claim 7 in section 9-3 above.

10. Claim 13 is rejected under 35 U.S.C. 103(a) as being unpatentable over Maddocks et al (Pub No US 2004/0201627 A1), supra, and Foster et al (US Patent No 6,684,260).

Claim 13

10-1. Regarding claim 13, Maddocks teaches the invention substantially as claimed. See section 6-5. Maddocks does not expressly teach a system wherein the computer operating system command list comprises at least one undo command that is each associated with an associated computer operating system command, wherein the command display displays the at least one undo command in association with the associated command, and wherein the command dispatcher causes execution of the at least one undo command that is associated with the selected at least one computer operating system command within the computer operating system command list.

Foster teaches a system wherein the computer operating system command list comprises at least one undo command that is each associated with an associated computer operating system command, wherein the command display displays the at least one undo command in association with the associated command, and wherein the command dispatcher causes execution of the at least one undo command that is associated with the selected at least one computer operating system command within the computer operating system command list. See section 7-1.

It would have been obvious to one of ordinary skill in the art, having the teachings of Maddocks and Foster before him at the time the invention was made, to modify the method for analyzing machine control sequences taught by Maddocks to include the undo commands of Foster, in order to obtain a method for analyzing a set of commands whereby analysis of commands can be more effectively run by allowing the ability to undo the execution of commands within the sequence. It would have been advantageous for one to utilize such a combination because as taught by Maddocks, "A need exists for a convenient and expedient way to view and analyze machine control sequences and the corresponding execution results" *[paragraph 4, lines 1-3].* By including the undo command, the user will be able to undo any commands that failed execution and more closely analyze the reason for such a failure. This would make it more convenient for the user because the user would be able to undo the command, make a change to correct the failure, and then execute the command again, instead of having to run the entire sequence repeatedly.

Conclusion

11. The prior art made of record on attached form PTO-892 and not relied upon is considered pertinent to applicant's disclosure. Applicant is required under 37 C.F.R. 1.111(c) to consider these references fully when responding to this action. The documents cited therein teach similar systems for the monitoring and analysis of operating system commands.

12. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Alvin H. Tan whose telephone number is 571-272-8595. The examiner can normally be reached on M-F.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John Cabeca can be reached on 571-272-4048. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

AHT
Assistant Examiner
Art Unit 2173

JOHN CABECA
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER